AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions and listings of claims in the application:

LISTING OF CLAIMS:

1. (currently amended): .A method for generating a test set for hard to detect faults, said method comprising:

a) identifying a set of hard to detect faults; and

b) generating the test set for the hard to detect faults by using an automatic test pattern generator;

wherein, the automatic test pattern generator comprises functionality for a global generator and is adapted to consider hardware overhead and test sequence lengths, said hardware ~~overheads~~ overhead being incurred when ~~each~~ a new testcube is added to the test set.

A1

2. (original): The method of claim 1, wherein the test set in step b is generated by a process further comprising:

(b) (i) calculating estimated cost using cost functions for each of the hard to detect faults;

(b) (ii) selecting a hitherto unselected target fault from the set of hard to detect faults that has a minimum cost;

(b) (iii) generating a testcube for the selected target fault;

(b) (iv) comparing real cost with estimated cost for the selected hard fault;

(b) (v) if the real cost is greater than a sum total of the estimated cost plus a predetermined error, performing an appropriate one of the following sub-steps:

(1) if there are still unselected faults in the set of hard to detect faults, returning to step (b) (ii); or

(2) if no unselected faults remain in the set of hard to detect faults, selecting a testcube having a minimum real cost;

(b) (vi) if real cost is not greater than a sum total of the estimated cost plus the predetermined error, retaining the test cube generated in step (b) (iii) as the selected test cube;

(b) (vii) marking all faults detected by the selected test cube; and

(b) (viii) adding the selected test cube into a current test set and updating a current generator.

3. (original): The method of claim 2 wherein said cost functions comprises of controllability costs, observability costs and test generation costs.

4. (original): The method of claim 2 wherein in step (b) (iii) a number of specified inputs for the testcube are minimized by bit stripping.

5. (original): The method of claim 3 wherein the controllability cost for an input is calculated based on the following formulae:

$$Cv(p_k) = \begin{cases} 0 & \text{if } g_k^i = U \\ 0 & \text{if } g_k^i = v \\ w & \text{if } g_k^i = \bar{v} \quad (\text{where } w \gg 1) \\ 1 & \text{if } g_k^i = X \text{ and } gg_k = B \\ 1 & \text{if } g_k^i = X \text{ and } gg_k = v \\ h & \text{if } g_k^i = X \text{ and } gg_k = \bar{v} \quad (\text{where } h \gg 1) \\ h & \text{if } g_k^i = X \text{ and } gg_k = N \end{cases} \tag{4}$$

where v is a binary value, 0 or 1, X is a don't care input,

Cv (p_k) represents cost for an input p_k,

g_k is an input in the current generator, and

gg_k represents an input in the global generator, and wherein

the controllability cost of each input is used to estimate a number of input conflicts and overriding signals that would be created by setting a line to a binary value v.

6. (original): The method of claim 3 wherein the controllability costs is calculated based on the following formulae:

a controllability cost for an internal circuit line *l* in the circuit, is

$$Cv(l) = \begin{cases} min_{l_a}\{Cc(l_a)\} & \text{if } v = c \oplus i \\ \sum_{l_a} C\bar{c}(l_a) & \text{otherwise,} \end{cases} \qquad (5)$$

where l_a and *l* are respectively the inputs and the output of a gate with controlling value *c* and inversion i.

7. (original): The method of claim 3, wherein the test generation cost is a sum of cost to activate a specific fault on a line and a cost of propagating the fault through the line.

8. (currently amended): A method of generating test sets for a fault list comprising hard to detect faults, the method comprising:

(a) initializing i ← 0 and *glob_gen* ← {*N,N,...,N)*;.

(b) initializing a current testcube set, $C^i \leftarrow \phi$;

(c) unmarking all faults in the fault list;

(d) initializing a current generator, $gen(^{Ci})$ = {X,X,….,X} and j ← 0;

(e) if there are no more faults in the fault list, then proceeding to step (m);

(f) generating a testcube c^j using an ATPG;

(g) adding the testcube c^j to a current testcube set, C^i <- C^i U c^j;

(h) mark faults detected by the testcube c^j;

(i) setting j <— j + 1;

(k) if the number of conflicting inputs of any testcube in C^i is greater than a positive integer M then $C^i \leftarrow C^I - c^j$, i ← i + 1, updating ~~the global generator~~ *global_gen*, and proceeding to step (b);

(l) if the number of conflicting inputs of any testcube in C^i is not greater than M, updating $gen(C^i)$ and proceeding to step (e);

(m) generating 3-weight weighted random pattern testing (WRPT) patterns by fixing inputs or applying pure random patterns to inputs according to $gen(C^i)$; and

(n) running fault simulation to drop the faults that are detected by the generated 3-weight WRPT patterns.

9. (original): The method of claim 8, further comprising:

(o) merging compatible overriding signals.

10. (currently amended): The method of claim 8, wherein said generating a test cube in step (f) is done by a process comprising:

(f) (i) calculating estimated cost using cost functions for each of the hard to detect faults;

(f) (ii) selecting a hitherto unselected target fault from the set of hard to detect faults that has a minimum cost;

(f) (iii) generating a testcube for the selected target fault;

(f) (iv) comparing real cost with estimated cost for the selected ~~hard~~ target fault;

(f) (v) if the real cost is greater than a sum total of the estimated cost plus a predetermined error, performing an appropriate one of the following sub-steps:

(1) if there are still unselected faults in the set of hard to detect faults, returning to step (f) (ii); or

(2) if no unselected faults remain in the set of hard to detect faults, selecting a testcube having a minimum real cost;

(~~b~~f) (vi) if real cost is not greater than a sum total of the estimated cost plus the predetermined error, retaining the test cube generated in step (f) (iii) as the selected test cube;

(f) (vii) marking all faults detected by the selected test cube; and

(f) (viii) adding the selected test cube into a current test set and updating the current generator.

11. (original): The method of claim 10, wherein inversely compatible inputs are merged by additionally inverting a signal before being fed to a gate.

12. (currently amended): The method of claim 10 wherein the controllability cost for an input is calculated based on the following formulae:

$$Cv(p_k) = \begin{cases} 0 & \text{if } g_k^i = U \\ 0 & \text{if } g_k^i = v \\ w & \text{if } g_k^i = \bar{v} \quad (\text{where } w \gg 1) \\ 1 & \text{if } g_k^i = X \text{ and } gg_k = B \\ 1 & \text{if } g_k^i = X \text{ and } gg_k = v \\ h & \text{if } g_k^i = X \text{ and } gg_k = \bar{v} \quad (\text{where } h \gg 1) \\ h & \text{if } g_k^i = X \text{ and } gg_k = N \end{cases} \tag{4}$$

where v is a binary value, 0 or 1, X is a don't care input,

Cv (~~pk~~ p_k) represents cost for an input ~~pk~~p_k,

~~gk~~ g_k is an input in the current generator, and

~~ggk~~ gg_k represents an input in the global generator, and wherein

the controllability cost of each input is used to estimate a number of input conflicts and overriding signals that would be created by setting a line to a binary value v.

13. (original): The method of claim 10 wherein the controllability costs is calculated based on the following formulae:

a controllability cost for an internal circuit line *l* in the circuit, is

$$Cv(l) = \begin{cases} min_{l_a}\{Cc(l_a)\} & \text{if } v = c \oplus i \\ \sum_{l_a} C\bar{c}(l_a) & \text{otherwise,} \end{cases} \tag{5}$$

where l_a and *l* are respectively the inputs and the output of a gate with controlling value *c* and inversion i.

14. (original): The method of claim 10, wherein the test generation cost is a sum of cost to activate a specific fault on a line and a cost of propagating the fault through the line.

15. (original): A parallel type test per scan built-in self test circuit comprising:

a circuit under test comprising inputs;

a set of scan flip flops that are connected to the inputs, each of said scan flip flops having at least a synchronous reset (R) or a synchronous preset (S) pin ;

a LFSR for loading random vectors that provide input to the set of scan flip-flops;

a decoder providing decoder outputs, wherein said decoder outputs control the R and S pins in the scan flip-flops, said decoder comprising a functionality of a global generator;

a counter providing inputs to the decoder that determine a state of the decoder outputs; and

an enable input for the decoder,

wherein the decoder provides overriding signals to by controlling the input to the R and S pins, said overriding signals overriding the random vectors based on tests in a generator for test patterns for hard faults, said tests being generated by an automatic test pattern generator.

16. (original): The circuit of claim 15 wherein the enable is provided by an AND gate, that performs an AND operation on a override enable input signal and a last scan input signal.

17. (original): The circuit of claim 16, wherein random patterns are overridden and the BIST enabled by providing a 1 input to the override enable input signal.

18. (original): The circuit of claim 16, wherein the last scan input signal is set to a 1 only at a last cycle of each scan shifting operation.

19. (original): The circuit of claim 15, wherein the counter being adapted to be set to 0 initially and maintained at 0 while a specific number (T) of random patterns are input, said random patterns being modified based on the generator provided by the decoder, said counter being further adapted to be incremented and T random patterns applied with a new generator, said increment being repeated until all generators have been applied by the decoder.

20. (original): The circuit of claim 15, wherein compatible overriding signals that can be merged are driven by a same output signal of the decoder, thereby reducing a number of decoder outputs.

21. (original): The circuit of claim 15, wherein inversely compatible overriding signals are driven by a decoder output directly and the same decoder output after passing through an invertor.

22. (original): The circuit of claim 15, wherein if a scan input is assigned a 1 by the global generator, then the corresponding scan flip-flop has an S pin.

23. (original): The circuit of claim 15, wherein if a scan input is assigned a 0 by the global generator, then the corresponding scan flip-flop has an R pin.

24. (original): The circuit of claim 15, wherein if a scan input is assigned both a 0 and 1 by the global generator, then the corresponding scan flip-flop has both R and S pins.

25. (original): The circuit of claim 15, wherein if a scan flip-flop already has a high active S pin and whose corresponding scan input is assigned a 1 in the global generator, then a two input OR gate is inserted between the S pin and a normal present signal.

26. (original): The circuit of claim 15, wherein if a scan flip-flop already has a high active R pin and whose corresponding scan input is assigned a 0 in the global generator, then a two input OR gate is inserted between the S pin and a normal present signal.

27. (original): The circuit of claim 15, wherein if a scan flip-flop already has a low active S pin and whose corresponding scan input is assigned a 1 in the global generator, then a two input AND gate is inserted between the S pin and a normal present signal.

28. (original): The circuit of claim 15, wherein if a scan flip-flop already has a low active R pin and whose corresponding scan input is assigned a 0 in the global generator, then a two input AND gate is inserted between the S pin and a normal present signal.

29. (original): A serial type test per scan built-in self-test circuit comprising:

a circuit under test comprising inputs;

a set of scan flip flops that are connected to the inputs;

a LFSR for loading random vectors that provide input to the set of scan flip-flops;

an AND gate and an OR gate inserted between said LFSR and the set of scan flip flops;

a decoder providing two decoder output signals D_0 and D_1, said decoder output signals being input to the AND and OR gates;

a generator counter that selects a generator providing inputs to the decoder;

a scan counter that provides input to the decoder, wherein a state of the decoder outputs are together determined by the counter input and the scan counter input; and

an enable input for the decoder,

wherein the decoder provides overriding signals, said overriding signals overriding the random vectors based on tests in a generator for test patterns for hard faults, said tests being generated by an automatic test pattern generator.

30. (original): The circuit of claim 29, wherein area overhead of the decoder is reduced by inserting toggle flip-flops between the two outputs of the decoder and inputs to the AND and OR gates.

31. (original): The circuit of claim 29, wherein inputs from the random vector corresponding to conflicting inputs in the generator are not overridden.

32. (original): The circuit of claim 29, wherein the scan_counter is adapted to increase by 1 at every positive edge of a scan clock and new values for scan inputs are scanned in serially until inputs corresponding to all pins have been scanned in,

wherein if a value in a generator is a 1, a 1 is scanned in as a scan input in a corresponding pin, instead of a value provided by the random vector, and

if a value in the generator is a 0, a 0 is scanned in as a scan input in a corresponding pin, instead of a value provided by the random vector and

if a value in the generator is a don't care or a conflicting value, then a value in the random vector is scanned in.

33. (original): The circuit of claim 30, wherein an order of scanning is determined by using genetic algorithms, wherein a permutation of scan elements in the scan chain is used as a genetic code, and the genetic algorithm is used to determine an order of scan elements that leads to a minimum number of minterms.

34. (original): The circuit of claim 33 wherein compatible overriding signals are merged prior to applying genetic algorithms.

35. (original): The circuit of claim 34, wherein scan inputs in a group of compatible scans are rearranged to satisfy routing or load capacity.